

**WHAT IS CLAIMED IS:**

1. A method for facilitating data synchronization, comprising the steps of:
  - (a) checking object store replica information corresponding to a remote object store;
  - (b) extracting a first set of objects to be synchronized with said remote object store;
  - (c) packing said first set of objects, their associated identifiers and synchronization versions into a request synchronization message;
  - (d) sending said request synchronization message to said remote object store;
  - (e) receiving a response synchronization message from said remote object store, said response synchronization message indicating a number of updated objects at the remote object store;
  - (f) resetting a corresponding set of synchronization versions to said updated objects; and
  - (g) purging off said updated objects.
2. The method of claim 1, further comprising the steps of:
  - (1) sending a request message to said remote object store if any information is missing from said object store replica information;
  - (2) receiving a response from said remote object store including a list of encoding methods; and
  - (3) registering said response in said object store replica information.
3. The method of claim 1, further comprising the steps of:
  - (1) updating objects based on said request synchronization message at said remote object store; and
  - (2) sending a response synchronization message providing a number of objects received and processed.
4. The method of claim 3, further comprising the step of:  
adding a list of encoding methods to said response synchronization message if said response synchronization message is a first message sent from said remote object store.

- 0  
9  
8  
7  
6  
5  
4  
3  
2  
1  
0
5. The method of claim 1, further comprising the step of:  
adding a field in said request synchronization message indicating whether said request synchronization message is a last request to said remote object store.
  - 5 6. A method for facilitating data synchronization, comprising the steps of:  
designing an application system comprising a network of devices;  
functionally dividing said application system into a set of primitive systems;  
determining at least one appropriate basic object store for each of said set of primitive systems; and
  - 10 10 replacing basic object stores that belong to multiple primitive systems by appropriate joint object stores.
  - 15 7. The method of claim 6, further comprising the steps of:  
receiving a first synchronization request from a first basic object store;  
15 updating an object based on said first synchronization request;  
receiving a second synchronization request from a second basic object store;  
updating said object based on said second synchronization request; and  
resolving any concurrent update conflicts in said object.
  - 20 8. The method of claim 7, wherein said step of resolving conflicts includes the steps of:  
initiating a first synchronization process with said first basic object store; and  
initiating a second synchronization process with said second basic object store.
  - 25 9. A method for facilitating data synchronization, comprising the steps of:  
exchanging update types and definitions among a set of object stores to commence a synchronization process;  
negotiating a data compression method among said set of object stores;  
comparing synchronization versions of said set of object stores;
  - 30 30 selecting a set of objects based on said comparing;  
transmitting said set of objects between said set of object stores; and  
transmitting meta objects associated with said set of objects between said set of object stores, said meta objects including a synchronization version and an identifier for each of said set of objects.

35

- 1000000000000000
10. The method of claim 9, wherein said selecting step includes the steps of:  
comparing objects in said set of object stores; and  
selecting objects representing differences between said set of object stores.
- 5        11. A method for facilitating data synchronization, comprising the steps of:  
recording information relating to a set of network links in a local database;  
determining an estimated average data transfer speed, round-trip transfer time,  
and packet size based on said information in said local database;  
selecting a flow protocol mode based on said determining;  
10        calculating a new packet size based on said determining; and  
dynamically adjusting said new packet size during a synchronization process.
- 15        12. The method of claim 11, wherein said step of dynamically adjusting includes  
the steps of:  
increasing said new packet size during said synchronization process if a data  
flow continues successfully for a period of time; and  
decreasing said new packet size during said synchronization process if said  
data flow fails within said period of time.
- 20        13. A computer program product for facilitating data synchronization, comprising:  
(a) logic code for checking object store replica information corresponding  
to a remote object store;  
(b) logic code for extracting a first set of objects to be synchronized with  
said remote object store;  
25        (c) logic code for packing said first set of objects, their associated  
identifiers and synchronization versions into a request synchronization  
message;  
(d) logic code for sending said request synchronization message to said  
remote object store;  
30        (e) logic code for receiving a response synchronization message from said  
remote object store, said response synchronization message indicating a  
number of updated objects at the remote object store;  
(f) logic code for resetting a corresponding set of synchronization versions  
to said updated objects; and  
35        (g) logic code for purging off said updated objects.

14. The computer program product of claim 13, further comprising:
- (1) logic code for sending a request message to said remote object store if any information is missing from said object store replica information;
  - (2) logic code for receiving a response from said remote object store including a list of encoding methods; and
  - (3) logic code for registering said response in said object store replica information.
- 5
15. The computer program product of claim 13, further comprising:
- (1) logic code for updating objects based on said request synchronization message at said remote object store; and
  - (2) logic code for sending a response synchronization message providing a number of objects received and processed.
- 10
16. The computer program product of claim 15, further comprising:  
logic code for adding a list of encoding methods to said response synchronization message if said response synchronization message is a first message sent from said remote object store.
- 15
17. The computer program product of claim 13, further comprising:  
logic code for adding a field in said request synchronization message indicating whether said request synchronization message is a last request to said remote object store.
- 20
18. A computer program product for facilitating data synchronization, comprising:  
logic code for designing an application system comprising a network of devices;  
logic code for functionally dividing said application system into a set of primitive systems;  
logic code for determining at least one appropriate basic object store for each of said set of primitive systems; and  
logic code for replacing basic object stores that belong to multiple primitive systems by appropriate joint object stores.
- 25
- 30
35. 19. The computer program product of claim 18, further comprising:

6  
5  
4  
3  
2  
1  
0  
-  
+  
=

logic code for receiving a first synchronization request from a first basic object store;

5 logic code for updating an object based on said first synchronization request;  
logic code for receiving a second synchronization request from a second basic object store;

logic code for updating said object based on said second synchronization request; and

logic code for resolving any concurrent update conflicts in said object.

10 20. The computer program product of claim 19, wherein said logic code for resolving conflicts includes:

logic code for initiating a first synchronization process with said first basic object store; and

15 logic code for initiating a second synchronization process with said second basic object store.

21. A computer program product for facilitating data synchronization, comprising:  
logic code for exchanging update types and definitions among a set of object stores to commence a synchronization process;

20 logic code for negotiating a data compression method among said set of object stores;

logic code for comparing synchronization versions of said set of object stores;

logic code for selecting a set of objects based on said comparing;

logic code for transmitting said set of objects between said set of object stores;

25 and

logic code for transmitting meta objects associated with said set of objects between said set of object stores, said meta objects including a synchronization version and an identifier for each of said set of objects.

30 22. The computer program product of claim 21, wherein said logic code for selecting includes:

logic code for comparing objects in said set of object stores; and

35 logic code for selecting objects representing differences between said set of object stores.

23. A computer program product for facilitating data synchronization, comprising:  
logic code for recording information relating to a set of network links in a local  
database;

5 logic code for determining an estimated average data transfer speed, round-trip  
transfer time, and packet size based on said information in said local database;

logic code for selecting a flow protocol mode based on said determining;

logic code for calculating a new packet size based on said determining; and

10 logic code for dynamically adjusting said new packet size during a  
synchronization process.

24. The computer program product of claim 23, wherein said logic code for  
dynamically adjusting includes:

15 logic code for increasing said new packet size during said synchronization  
process if a data flow continues successfully for a period of time; and

logic code for decreasing said new packet size during said synchronization  
process if said data flow fails within said period of time.

20

25

30

35